Preface

The origins of cryptography are rooted in antiquity but in the last 40 years the subject has experienced explosive growth which has led to deep changes both in its foundations and methodology, and also in the way it is used in the real world, where many new applications have arisen that were not even dreamed of in the middle of the twentieth century.

On the foundational side, we have seen the emergence of key concepts, such as that of one-way function, which now occupies a central role not only in public-key cryptography, where it originated, but also in private-key cryptography. Methodologically, randomness and complexity have surged to the forefront and, with their help, security has been rigorously defined in such a way that makes it possible to obtain reductionist proofs which show that some cryptographic schemes are secure on the assumption that certain mathematical problems are computationally hard. These "security reductions" do not guarantee the security of a scheme in an absolute sense because, in addition to the fact that no computational problems have been proven to be hard so far, the security definitions are formalizations that may not capture all the real-world complexities of the concept and there are other issues that have to be taken into account (implementation issues, side-channel attacks…). However, the reductions put the security of cryptographic schemes on firmer ground since they make explicit the relation with well-studied problems that are widely believed to be hard.

In addition, cryptography is now well beyond its original goal of transmitting information secretly and, to mention just one example, it has been argued that authentication, as provided by digital signatures, is even more important than the secrecy provided by encryption schemes. On top of all this, cryptography is currently being used by almost everyone on a daily basis when, say, connecting to a WiFi network, using a Web browser to make a secure connection, making a payment with a credit card, and so on. This has led to the development of cryptographic schemes that, besides enjoying good security properties, are sufficiently efficient for real-world use.

This book is an introduction to cryptography that, in addition to discussing the relevant theoretical constructions, employs a programing-oriented approach to

study the most important cryptographic schemes in use today, as well as the main cryptanalytic attacks against these schemes. It is based on undergraduate courses in both Mathematics and Computer Science, taught at the University of Santiago de Compostela, where only a fraction of the contents included here was covered. The book is aimed at advanced undergraduate or beginning graduate students, the main prerequisite being a basic knowledge of linear algebra and elementary calculus. In addition, some knowledge of probability and abstract algebra—modular arithmetic, groups, polynomials, fields, etc.—would be helpful but is not required and the necessary material is included in the text.

The focus of this book is on cryptographic algorithms and the cryptographic schemes built from them, as well as on the cryptanalytic algorithms used to attack these schemes, rather than on the higher level interactive protocols that require the cooperation of at least two parties for their execution. We treat the Diffie–Hellman key agreement protocol in detail in Chap. 7, because of its historical and practical importance, and we also occasionally give a brief description of other protocols, but we refer to [65, Chap. 4] for a good introduction to several important classes of interactive protocols, including protocols for user identification, for electronic voting, and for digital cash. More detailed descriptions of some of these and other protocols are given in [172, 134, 189] and in the more theoretically oriented reference [99].

This book includes cryptanalytic algorithms even when the goals of cryptanalysis are, in principle, opposite to those of cryptography because, while the latter tries to design secure schemes, cryptanalysis tries to break them. Despite these conflicting goals, cryptanalysis is one of cryptography's best friends. Checking that a cryptographic scheme resists the known cryptanalytic attacks may serve to increase confidence in its security properties and, even when the scheme has a security reduction to some presumably hard problem, attacks on this problem provide the tools to evaluate security in practical terms and to estimate, for example, which key sizes should be used. In addition to this, new attacks are constantly being developed—and existing attacks are being refined—and the possibility that some of them might not be prevented by the standard security definitions should not be discarded.

A distinctive feature of this book is the inclusion of Maple implementations of most cryptographic schemes—and also of the main cryptanalytic attacks—mentioned in the text. These implementations are not limited to the basic algorithms underlying the schemes, i.e., to the cryptographic primitives, but include all the relevant algorithms that build up each scheme (for example, depending on the kind of scheme, key generation, encryption, decryption, signing, verification) and are able to deal with real messages given either as text strings or files using keys of realistic size. This allows us to include numerous examples which, together with the Maple code, are an integral part of the text and serve to illustrate the theoretical constructions.

Frequently, cryptography books that rightfully warn the reader against the dangers of plain RSA (or "textbook RSA" as it is often called; see Sect. 8.3 for the reasons why it is insecure), when poised to give a concrete RSA example present …

what? … plain RSA, of course, and often with a 4- or 5-digit modulus that, paraphrasing Knuth, would have been factored by Fermat in minutes—if not in seconds—on the back of an envelope.

While appearing somewhat paradoxical, this is fair enough. The example serves to illustrate the weaknesses of the basic construction and may be taken as a starting point from which the reader can imagine what the real construction would look like. Moreover, small-sized examples can be manipulated by hand and, by using them, quite a few lines of text are spared. Even assuming the use of a computer, it would not be realistic to try to include full real-world examples in an introductory textbook, for there are myriads of technical details that would have to be taken into account and that would probably obscure the basic important concepts with the danger that the reader would not see the forest for the trees. In addition, one usually has to deal with some resource limitations; for example, in our setting it is virtually impossible to use Maple to obtain the random seeds required by many schemes.

We feel, however, that there is real added value in giving examples that go beyond the very basic algorithms and that also have realistic size. While they are still simplified models, they come much closer to the look and feel of real-world cryptography and, in our experience, students like them better because they serve to encrypt real messages and, in contrast to the toy examples mentioned, cannot be broken by anyone with access to a computer in a matter of seconds. Also, many crucial details are lost in the examples when one limits them to dealing with cryptographic primitives. For instance, in the case of RSA, we can use an implementation of RSA-OAEP to help appreciate that a secure—in the reductionist sense—version of RSA is much more than the modular exponentiation that is embodied in the RSA primitive.

We must warn the reader that the Maple implementations of cryptographic schemes given here are not intended for standard cryptographic use but, rather, for the purpose of experimenting and learning. In several cases, the value of these implementations stems precisely from the fact that they let us to better appreciate the reasons why some schemes are insecure, but, even when the underlying schemes have a reductionist security proof, there may be some missing ingredients in the Maple version. For example, many of these functions require as input random keys or seeds that cannot be generated within Maple. However, assuming that some additional conditions—like having access to a source of true randomness—hold, these implementations may give a reasonably simple approximation to the real thing within the more restricted Maple environment.

The Maple code and the programming examples try to build a bridge between the world of theoretical cryptography, where security is the main concern, and the world of practitioners and implementers where efficiency is often the focus, sometimes even at the expense of security. We hope that some of these examples will help to show that schemes with good security properties are not much difficult to understand and implement than other less secure schemes.

The Maple constructions given in the text are not only limited to cryptographic schemes but also include number-theoretic algorithms used in many cryptographic

primitives, as well as some of the most important cryptanalytic algorithms. The latter includes, for example, algorithms for factoring integers and for solving discrete logarithms. Since these algorithms are generally much less efficient than those involved in the implementation of the schemes, it is difficult to give examples of realistic size in this case but we try, nonetheless, to present examples that allow the reader to experiment and get a feeling for the behavior of the algorithms in real-world situations. The use of a computer algebra package greatly facilitates the implementation of schemes and algorithms because many of the mathematical and, more specifically, number-theoretic algorithms underlying these constructions are already built-in and immediately available to us. For example, algorithms for modular exponentiation or extraction of modular square roots are extensively used in cryptography and we discuss them in the text but, when they are required to implement other algorithms, we use the versions in Maple's kernel, which are fast and convenient.

The choice of Maple as a programming environment was prompted by the fact that this was the computer algebra system used in the courses which provided the initial motivation for the book but there are several other systems that could be used perfectly well for the same purpose. Among them, we mention the open source computer algebra package Sage (downloadable from [171]) which integrates a set of about 100 open source packages under a common interface and is emerging as a very good alternative to the more established systems. Other computer algebra systems with the capability to develop all these examples are Magma and Mathematica. In principle, an instructor familiar with any of these systems should not have much trouble to translate the Maple functions[1] included in the text into an alternative language. Except for some input–output and conversion routines, as well as some built-in functions which are specific to a given environment, the bulk of the programs rely only on basic programing constructions, such as loops, which have a similar structure in different programing languages. It is clear that having some programing background—as will be the case with Computer Science students and also with most Mathematics or Engineering students—will allow readers to modify the programs included or to write their own programs but we feel that, even in the absence of this background, some minimal guidance will be sufficient to explore the examples and to play and experiment with them. In addition, we refer to the extensive help facilities included in the Maple package, to the online help page at the Maplesoft web site [135] which also contains introductory and advanced programing guides, and to the many books of introduction to Maple, among which we specifically mention [100].

The structure of the book generally proceeds by discussing some cryptographic problem—and, in particular, the all-important security aspects—presenting the algorithms involved and then giving Maple implementations and examples of

[1] The term *function* is used in this book with two different meanings. One of them refers to the standard mathematical concept and the other refers to a *Maple function* and is essentially a synonym for *Maple procedure*, so that no confusion should arise.

these algorithms. Thus, the examples are not relegated to the end but are an integral part of the text flow. Although it is perfectly possible to read the book skipping the programing sections, this is far from ideal and we recommend going through them if at all possible. On the other hand, we have tried to include proofs of all the basic results but, when dealing with some more advanced results, we sometimes give only a proof outline or even, in other cases, a reference to an appropriate source. This is more often the case in relation with theoretical security results because the focus of this book is on algorithms rather than on these results. In all cases, however, we try to present the main ideas on which the proofs are based and to discuss them in context, and in these discussions we include, where appropriate, abundant references to the literature.

The contents of the book may be summarized as follows. After presenting, in the Introduction, the basic ideas that serve to define cryptography and the associated concepts of cryptanalysis and cryptology, we give in Chap. 1 an introduction to classical ciphers that requires little background and serves to introduce the basic cryptographic and cryptanalytic techniques and to appreciate some conditions that an encryption scheme must satisfy to meet even minimal security requirements. Chapter 2 provides the necessary background to read most of the rest of the book, with special attention devoted to modular arithmetic and number-theoretic algorithms that are extensively used throughout.

Chapter 3 gives an introduction to private-key cryptography, discussing perfect secrecy and the one-time pad and introducing one-way functions and pseudorandom number generators, as well as private-key encryption schemes and the important notions of CPA and CCA security. Then, in Chap. 4, block ciphers and their modes of operation are studied, with special attention devoted to AES, the Advanced Encryption Standard, which is implemented in Maple along with some of the more interesting modes of operation. Chapter 5 is devoted to message authentication and discusses several kinds of message authentication codes (MACs) as well as cryptographic hash functions. It includes Maple implementations of CMAC, GCM/GMAC, SHA-256 and HMAC, and also a simulation of the birthday attack.

Chapter 6 is entirely devoted to a study of more advanced number-theoretic algorithms that are relevant for cryptography, including algorithms for primality testing and prime generation, as well as some of the most important algorithms for the integer factorization problem—such as the quadratic sieve, discussed with the help of Maple—and the discrete logarithm problem.

In Chap. 7 we introduce the basic ideas behind public-key cryptography and discuss the Diffie–Hellman protocol and the hard problems underlying it, as well as some cryptanalytic attacks. Chapter 8 is devoted to public-key encryption and, after introducing the appropriate security notions for this setting, some of the most important public-key encryption schemes are studied. They include RSA, Rabin, and Elgamal, and the more secure variants RSA-OAEP, Rabin-SAEP$^+$, and Cramer–Shoup, as well as Paillier encryption. The reductionist security proof of Cramer–Shoup is presented and Maple implementations of all these schemes are given. Chapter 9 is devoted to the study of digital signatures, including RSA and

Elgamal signatures and the Digital Signature Algorithm (DSA), as well as schemes with security reductions, such as the PSS—also implemented in Maple—and Cramer–Shoup signatures. This chapter ends with a discussion of public-key infrastructures and certificates.

In Chap. 10 we introduce the recent subject of identity-based (IB) cryptography, including IB signature schemes and IB encryption (IBE) and give an implementation of the Cocks IBE scheme as well as a description of the Boneh–Franklin IBE scheme. The final chapter is devoted to an introduction to elliptic curve cryptography and includes a discussion of the elliptic curve discrete logarithm problem and related algorithms, as well as a study of elliptic curve-based signatures and encryption. This study is carried out with the help of Maple which is also used to implement the Elliptic Curve Digital Signature Algorithm, ECDSA. Appendix A contains some useful Maple conversion functions which are used throughout the text.

As can be seen from the summary above, we have included recent developments such as secure schemes (relative to some hard problem) and identity-based cryptography and, although we have not included a study of some advanced subjects such as, for example, fully homomorphic encryption or elliptic curve pairings and their cryptographic applications, we feel that someone who has read the book is well equipped to proceed with this study.

Finally, let us note that, as already mentioned, most of the schemes studied as well as many of the algorithms used for cryptanalytic attacks are implemented in Maple, giving the reader the opportunity to experiment with them. The implementations of schemes often rely on the standards contained in NIST's (the US National Institute of Standards and Technology) and other bodies' publications and, in the case of the more recent schemes, in the papers describing them. The Maple code is available from the book's website:

http://sites.google.com/site/cryptomaple

and includes, for each chapter of the book, a Maple language file called ChapterXX.mpl and a Maple worksheet called ChapterXX.mw. The language file contains all the Maple code pertaining to the chapter in question and can be read from the corresponding worksheet which contains all the chapter examples; other examples that will work with this code may be easily generated. All the code included should work in Maple 13 or later and, in fact, most of it will work also in Maple 11 and probably also in earlier versions, the exception being some functions that make use of the element-wise operator '~' (used to distribute an operation over a data structure such as a list or an array). The use of this operator may be avoided by using the commands 'map' or 'zip' instead and this can be an exercise for the reader as has been pointed out at appropriate places in the text. The book Web page also contains a list of errata and the author would be grateful for any comments or corrections, which may be sent to the email address: gomez.pardo@usc.es.

Suggestions for course organization and teaching. This book contains more than enough material for developing several kinds of courses depending on course duration and background requirements and offering different course orientations and flavors. We outline here some of these possibilities:

- A basic introductory course with the minimal background requirements mentioned above and that could have two variants according to course length. For a one-semester course, the material to be covered could be: a brief discussion of classical ciphers and their shortcomings as presented in Chap. 1, skipping some of the cryptanalytic details; a review of asymptotic notation, the basic number-theoretic algorithms (Euclidean algorithm, Euler's theorem, modular exponentiation), and the construction of finite fields, taken from Chap. 2; most of Chap. 3 (skipping some of the more advanced material); the description of AES and the modes of operation from Chap. 4; the basic properties of MACs and the construction of CBC-MAC/CMAC (Sects. 5.1–5.3); hash functions (Sect. 5.6 through Sect. 5.6.3 and Sect. 5.7); the basics of primality testing (Sect. 6.2) and a quick overview of factorization and discrete log algorithms extracted from the first subsections of Sects. 6.4 and 6.5; an introduction to public-key cryptography (Chap. 7) focusing on the DH protocol and the DH problems; the basics of public-key encryption, including the definitions in Sects. 8.1 and 8.2, a discussion of RSA after Sect. 8.3 (through Sect. 8.3.5); and, finally, digital signatures with the material in Sects. 9.1 and 9.2, the first part of Sect. 9.3 including hashed RSA signatures, and Sect. 9.7. For a longer two-semester introductory course, some of the preceding topics would be studied in greater detail and, moreover, the following additional material could be covered: Sects. 5.4, 6.3, 8.3, 8.5, 9.3, 9.4, and 9.6, and a selection of topics from Chap. 11 up to a discussion of the advantages of elliptic curve cryptography.
- A theoretically oriented course whose focus is on the precise definition of security concepts and on the cryptographic schemes which have reductionist security proofs. The precise contents for this course might vary according to the students' background but, in addition to some of the material mentioned for the basic course, could also include Sects. 5.4 and 5.5 on authenticated encryption, Sects. 8.4 and 8.6 on CCA secure encryption schemes, Sect. 8.8 on homomorphic encryption, Sect. 9.5 on CMA secure signatures, and, possibly, an introduction to indentity-based cryptography as presented in Chap. 10. 'Theoretically oriented' here does not mean that practical aspects would not be considered and, in particular, the practical challenges involved in implementing secure schemes could be explored with the help of some of the Maple implementations included.
- A mathematically oriented course which emphasizes number-theoretic ciphers and their cryptanalysis. Starting with the basics from Chap. 2, it would include some material from Chap. 3 (one-way functions and pseudo-random generators) and Chap. 5 (hash functions) and then focus on the study of number-theoretic algorithms (including most of Chap. 6) and their applications to building and

attacking public-key cryptosystems in Chap. 7–9, (specifically, Diffie–Hellman, RSA, Rabin, Elgamal, Goldwasser–Micali and Paillier) as well as a study of elliptic curve cryptography based on Chap. 11.

- A practice-oriented course requiring little mathematical background and with emphasis on applications and, specifically, on the Maple implementations of schemes that are defined in several standards, in particular, AES plus modes of operation from Chap. 4, CMAC, GCM/GMAC, SHA-256 and HMAC from Chap. 5, RSAES-OAEP from Chap. 8, RSASSA-PSS from Chap. 9 and ECDSA from Chap. 11.
- A mathematically advanced course addressed to students with a strong mathematical background, with focus on public-key cryptography and on cryptanalysis. This would include material from Chap. 3 on one-way functions and pseudorandom generators and from Chap. 5 on hash functions, as well as the contents in Chaps. 6–11.

Acknowledgments

Part of this book was written while I was on sabbatical leave from the University of Santiago de Compostela in 2009–2010. I thank the University for granting this leave and, in particular, my colleagues at the Algebra Department for being generous and making it possible, as well as for their support during the preparation of the manuscript.

I also acknowledge the support provided by Project 04555/GERM/06 from Fundación Séneca of the Región de Murcia (partially financed by FEDER funds from the European Union), and by Project INCITE09E2R207117ES from Xunta de Galicia.

I would like to thank José Luis García and Ángel del Río for their genuine interest in the book and for their valuable corrections and suggestions.

I also thank Ronan Nugent and Ulrike Stricker-Komba at Springer for their assistance and support through the publication process.

Finally, I wish to thank my wife, Nieves, and my son, Carlos, for reading parts of the book and offering helpful suggestions, as well as for their support and understanding during the long hours devoted to this project.

Santiago de Compostela, October 2012 José Luis Gómez Pardo

http://www.springer.com/978-3-642-32166-5

Introduction to Cryptography with Maple

Gómez Pardo, J.L.

2013, XXX, 706 p.,
ISBN: 978-3-642-32166-5